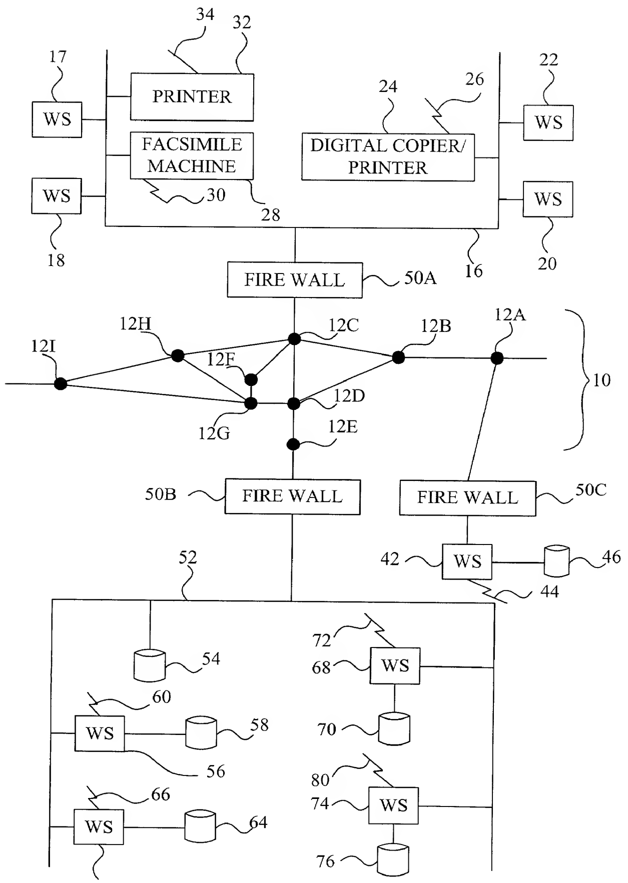

34
32
17
22
24
26
PRINTER
WS
WS
FACSIMILE MACHINE
DIGITAL COPIER/ PRINTER
WS
WS
30
28
18
16
20
FIRE WALL
50A
12H
12C
12B
12A
12I
12F
10
12D
12G
12E
50B
FIRE WALL
FIRE WALL
50C
52
42
WS
46
44
72
68
WS
54
60
70
58
WS
56
80
66
74
WS
64
WS
76
62

FIG. 1

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 2 OF 32

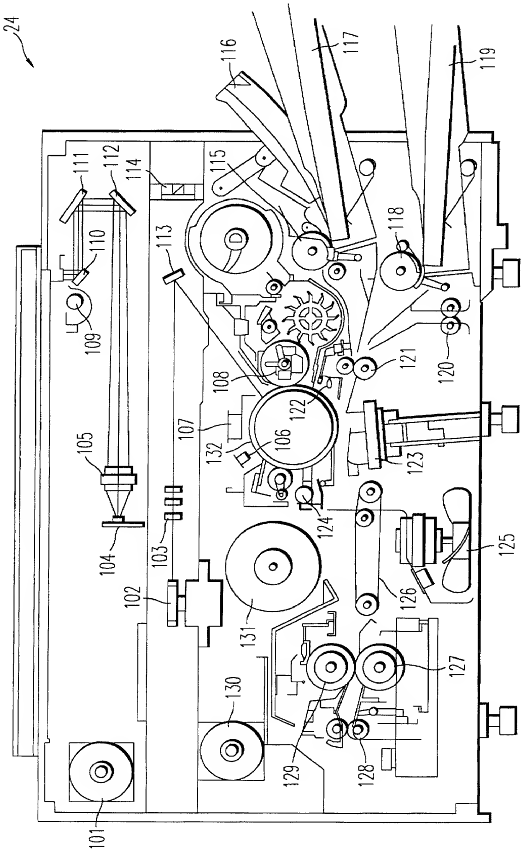


FIG. 2

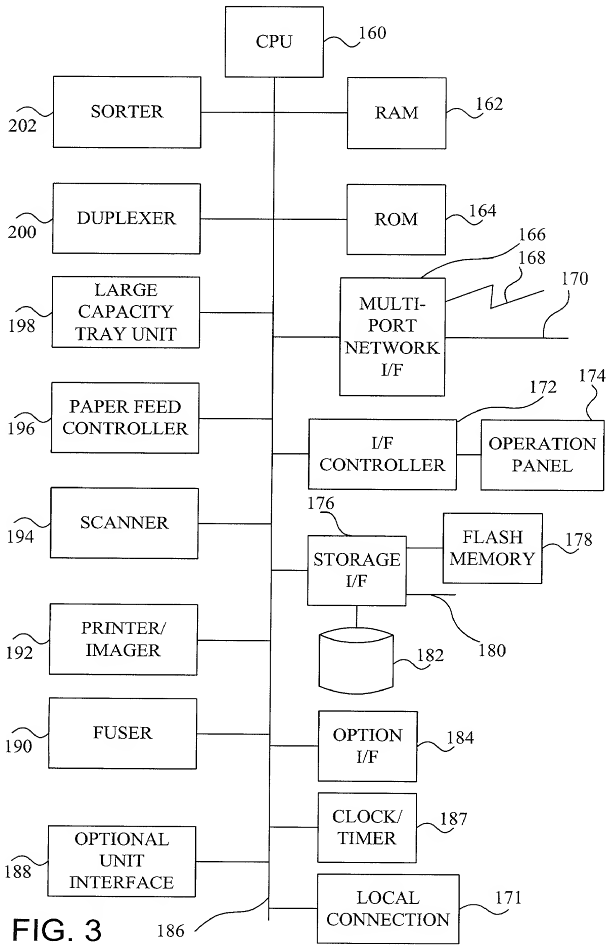

CPU
160
SORTER
202
RAM
162
DUPLEXER
200
ROM
164
166
168
170
LARGE CAPACITY TRAY UNIT
198
MULTI-PORT NETWORK I/F
PAPER FEED CONTROLLER
196
172
174
I/F CONTROLLER
OPERATION PANEL
SCANNER
194
176
STORAGE I/F
FLASH MEMORY
178
PRINTER/ IMAGER
192
182
180
FUSER
190
OPTION I/F
184
CLOCK/ TIMER
187
OPTIONAL UNIT INTERFACE
188
LOCAL CONNECTION
171
186

FIG. 3

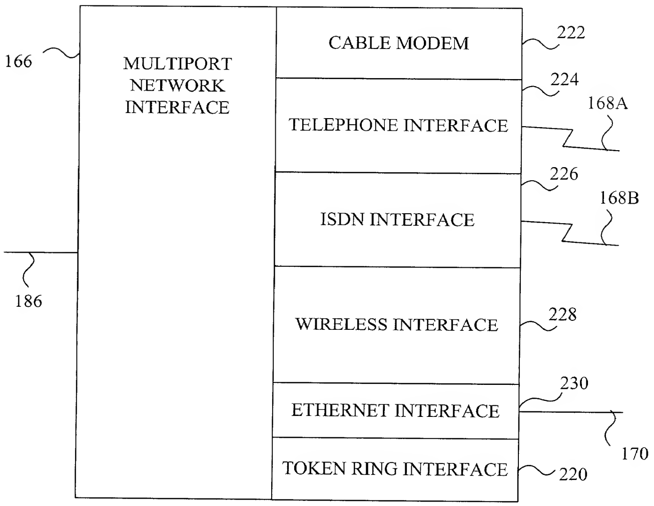

MULTIPORT NETWORK INTERFACE
CABLE MODEM
TELEPHONE INTERFACE
ISDN INTERFACE
WIRELESS INTERFACE
ETHERNET INTERFACE
TOKEN RING INTERFACE
166
186
222
224
168A
226
168B
228
230
170
220

FIG. 4

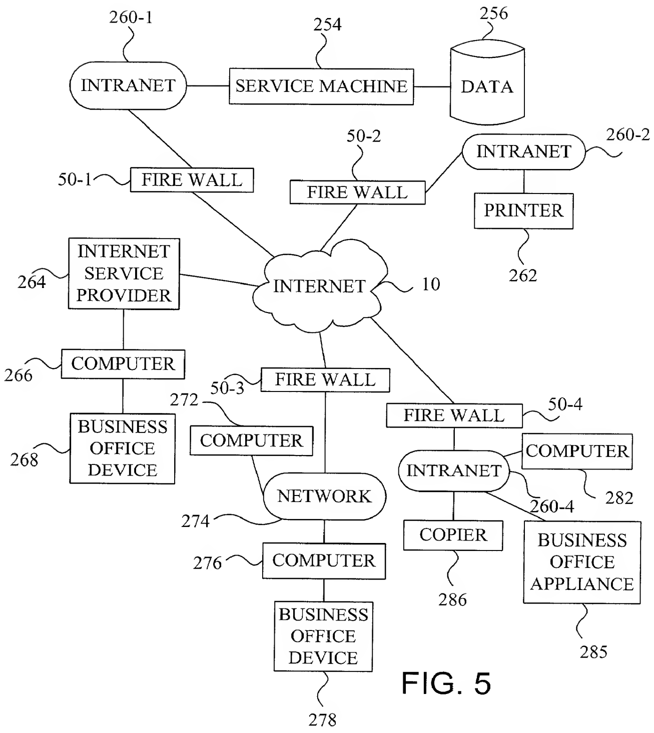

260-1
254
256
INTRANET
SERVICE MACHINE
DATA
50-2
260-2
INTRANET
50-1
FIRE WALL
FIRE WALL
PRINTER
INTERNET SERVICE PROVIDER
264
INTERNET
10
262
COMPUTER
266
FIRE WALL
50-3
272
50-4
FIRE WALL
BUSINESS OFFICE DEVICE
268
COMPUTER
COMPUTER
INTRANET
282
NETWORK
260-4
274
COPIER
BUSINESS OFFICE APPLIANCE
276
COMPUTER
286
BUSINESS OFFICE DEVICE
285
278

FIG. 5

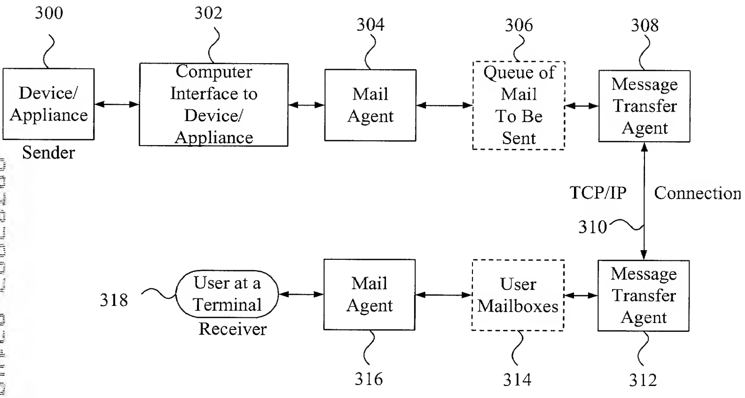

300
Device/ Appliance
Sender
302
Computer Interface to Device/ Appliance
304
Mail Agent
306
Queue of Mail To Be Sent
308
Message Transfer Agent
TCP/IP
Connection
310
318
User at a Terminal
Receiver
316
Mail Agent
314
User Mailboxes
312
Message Transfer Agent

FIG. 6A

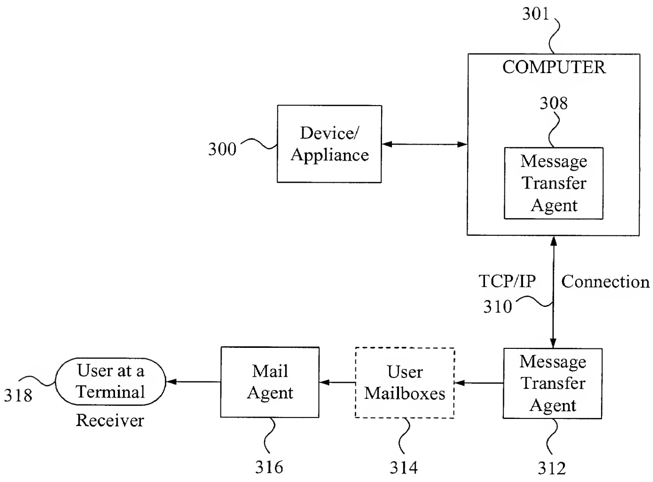

301
COMPUTER
308
Message Transfer Agent
300
Device/ Appliance
TCP/IP
Connection
310
318
User at a Terminal
Receiver
Mail Agent
User Mailboxes
Message Transfer Agent
316
314
312

FIG. 6B

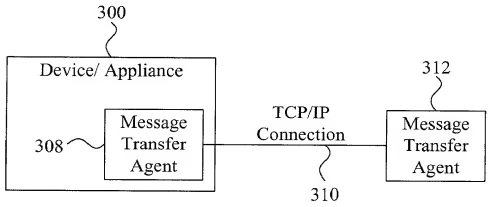

300
Device/ Appliance
308
Message Transfer Agent
TCP/IP Connection
310
312
Message Transfer Agent

FIG. 6C

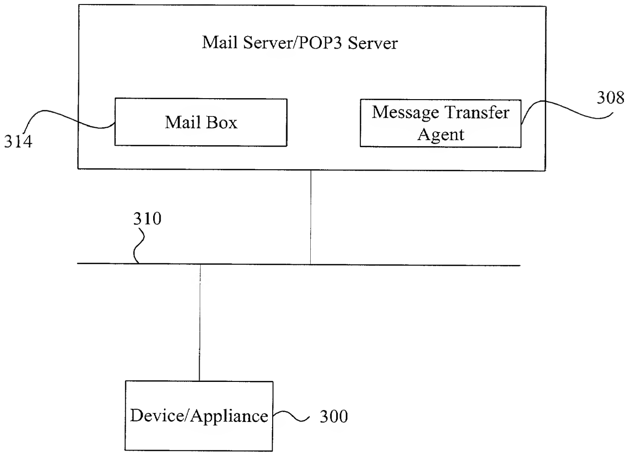

Mail Server/POP3 Server
Mail Box
Message Transfer Agent
314
308
310
Device/Appliance
300

Fig. 6D

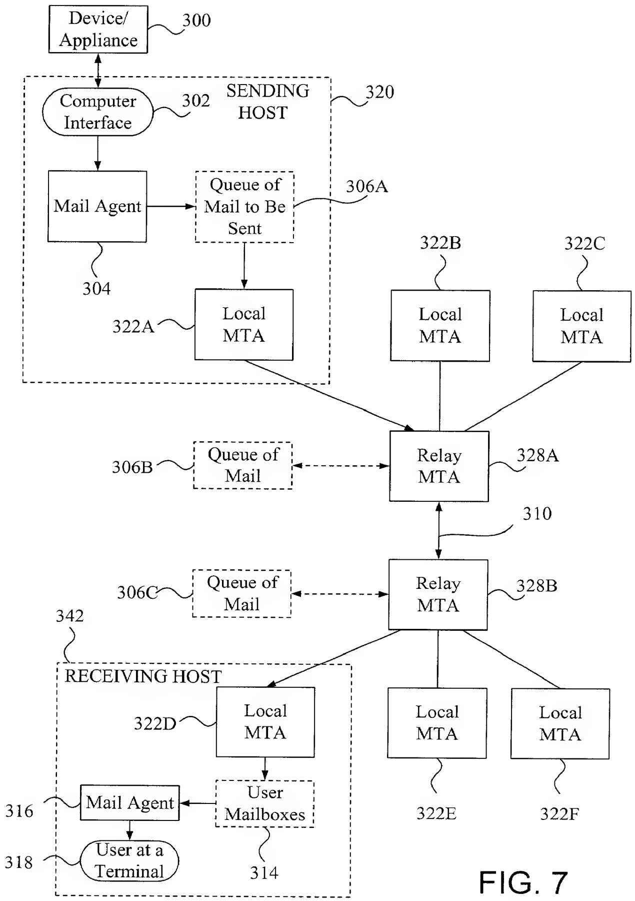

Device/ Appliance
300
Computer Interface
302
SENDING HOST
320
Mail Agent
304
Queue of Mail to Be Sent
306A
Local MTA
322A
322B
Local MTA
322C
Local MTA
Relay MTA
328A
Queue of Mail
306B
310
Relay MTA
328B
Queue of Mail
306C
342
RECEIVING HOST
Local MTA
322D
Local MTA
322E
Local MTA
322F
User Mailboxes
314
Mail Agent
316
User at a Terminal
318

FIG. 7

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 10 OF 32

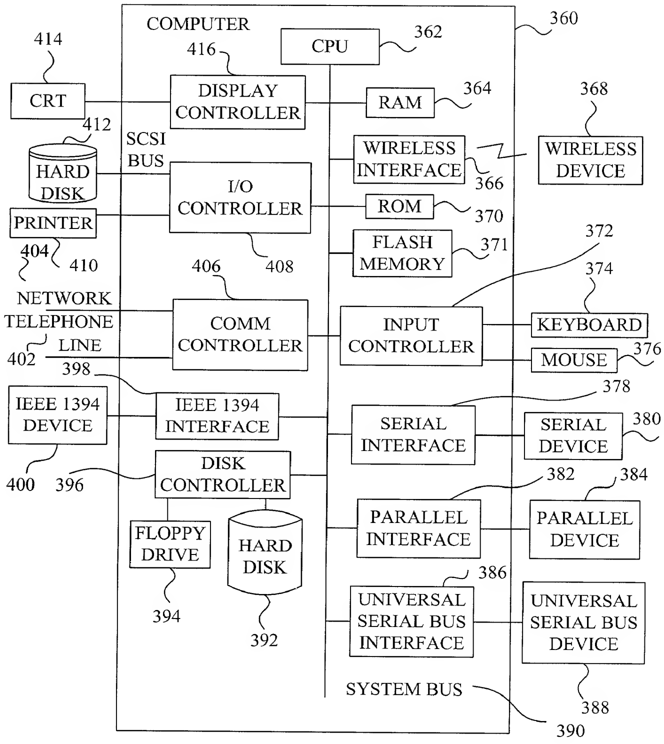


FIG. 8

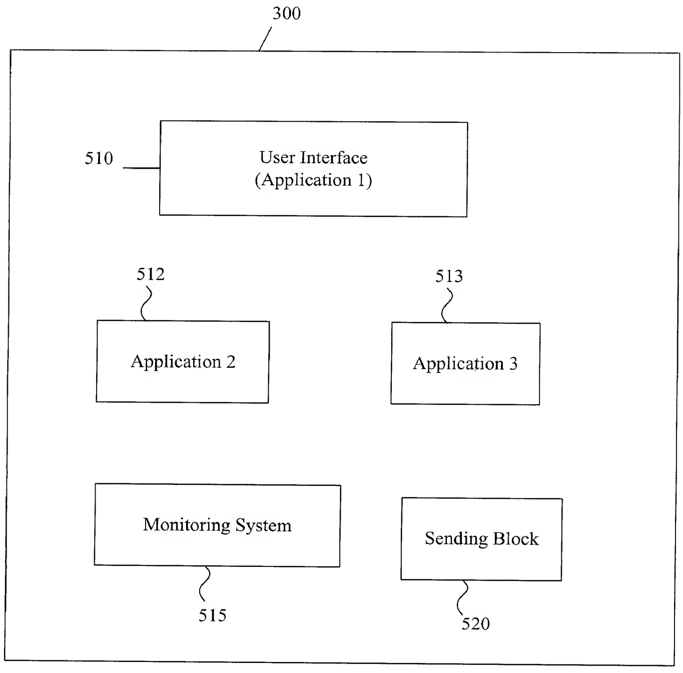

300
510
User Interface
(Application 1)
512
Application 2
513
Application 3
Monitoring System
515
Sending Block
520

FIG. 9

FIG. 10

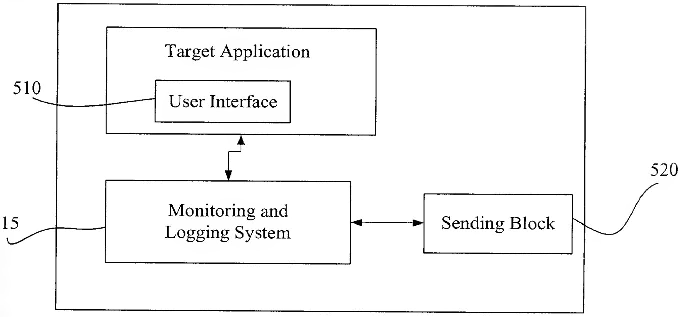
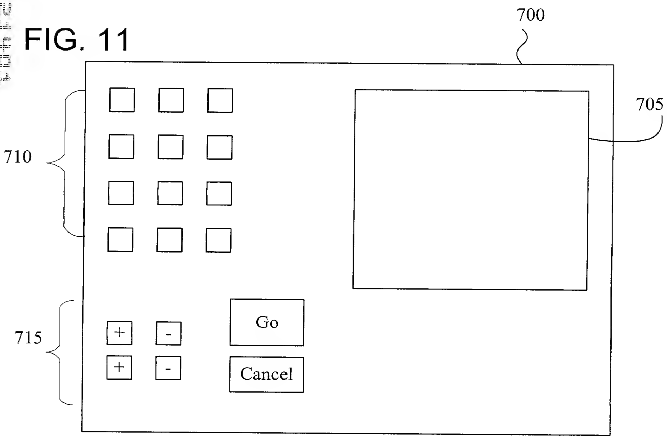

Target Application
510
User Interface
Monitoring and Logging System
515
Sending Block
520

FIG. 11


700
705
710
715
+
-
+
-
Go
Cancel

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 13 OF 32

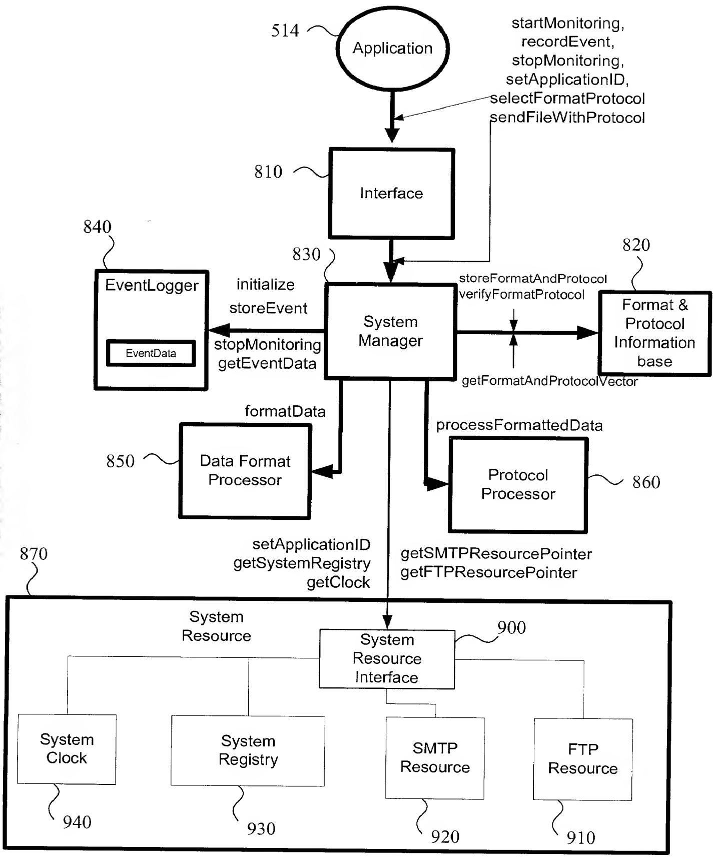


FIG. 12A

Return Value	Function Name	Description
bool	getNextSession	Returns false when there is no more session; true otherwise
string	getFileName	Returns file name for the EventData
map<string, string>	getSessionInformation	Returns the map. Keys are UserID, ApplicationID, CumulativeSessionNumber, StartTime, and Duration.
map<string, vector<string>>	getSessionEventData	Returns the map. Keys are EventName and EventTiming. The values of EventTiming vector are in the unit of 10th of a second converted from unsigned integer to string.

FIG. 12B

Return Value	Function Name	Description
bool	getNextLine	Returns one line of string data as an out parameter string. The function returns true if there is a line; false if no more line exists with empty string.
string	getFileNameWithSuffix	Returns file name for the data with suffix if applicable
enum	getDataType	Returns the data type, BINARY or TEXT

FIG. 12C

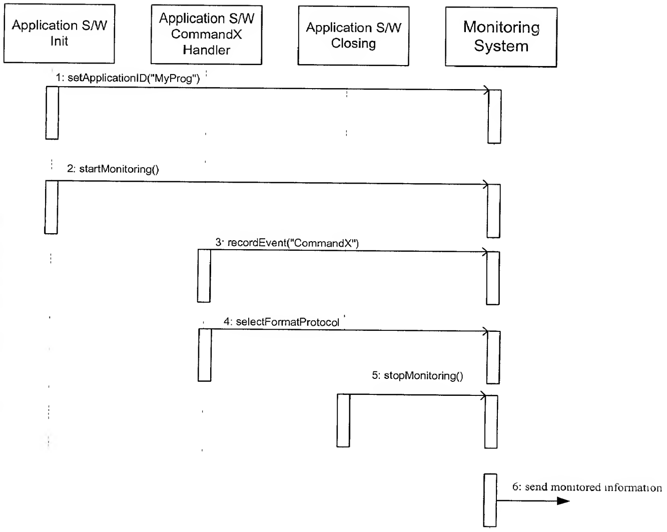

Application S/W Init
Application S/W CommandX Handler
Application S/W Closing
Monitoring System
1: setApplicationID("MyProg")
2: startMonitoring()
3: recordEvent("CommandX")
4: selectFormatProtocol
5: stopMonitoring()
6: send monitored information

FIG. 13A

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 17 OF 32

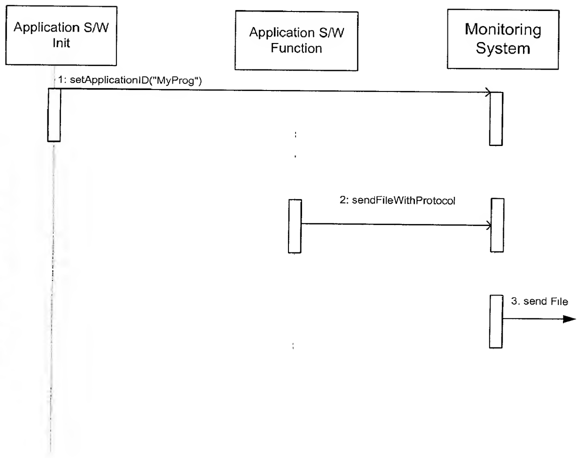
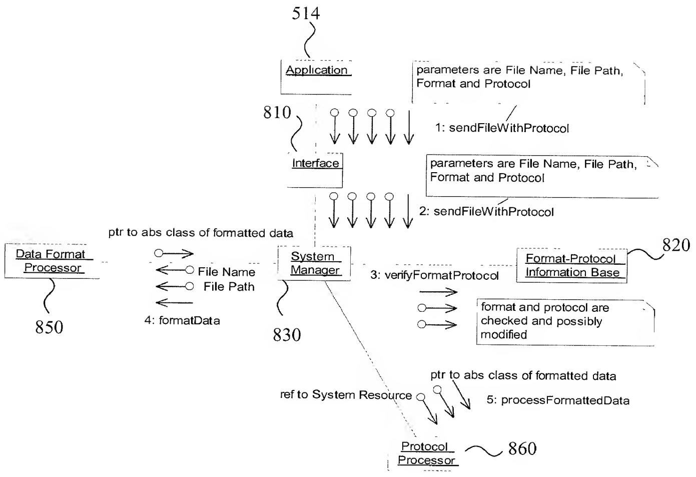


FIG. 13B

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 18 OF 32



514
Application
parameters are File Name, File Path, Format and Protocol
810
1: sendFileWithProtocol
Interface
parameters are File Name, File Path, Format and Protocol
2: sendFileWithProtocol
ptr to abs class of formatted data
Data Format Processor
File Name
File Path
4: formatData
850
System Manager
830
3: verifyFormatProtocol
Format-Protocol Information Base
820
format and protocol are checked and possibly modified
ptr to abs class of formatted data
ref to System Resource
5: processFormattedData
Protocol Processor
860

FIG. 14

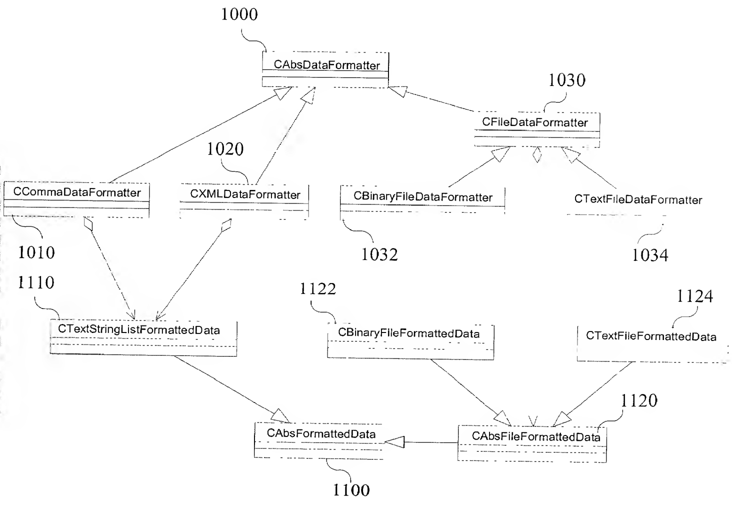

1000
CAbsDataFormatter
1030
CFileDataFormatter
1020
CCommaDataFormatter
CXMLDataFormatter
CBinaryFileDataFormatter
CTextFileDataFormatter
1010
1032
1034
1110
1122
1124
CTextStringListFormattedData
CBinaryFileFormattedData
CTextFileFormattedData
1120
CAbsFormattedData
CAbsFileFormattedData
1100

FIG. 15

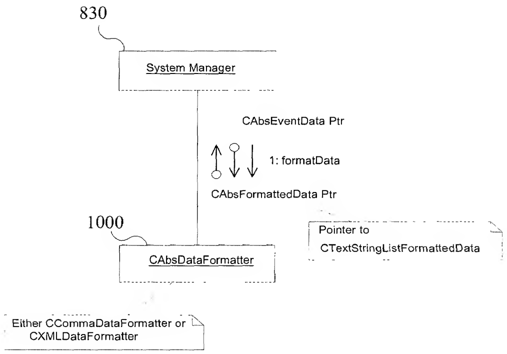

830
System Manager
CAbsEventData Ptr
1: formatData
CAbsFormattedData Ptr
1000
CAbsDataFormatter
Pointer to CTextStringListFormattedData
Either CCommaDataFormatter or CXMLDataFormatter

FIG. 16

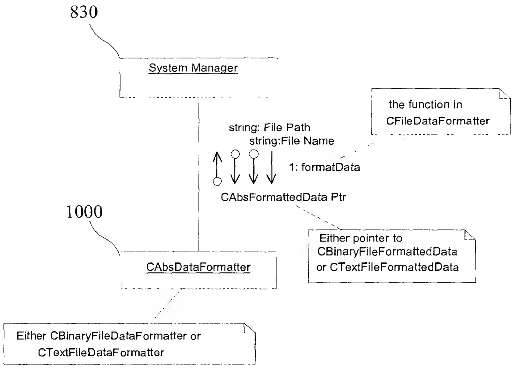

830
System Manager
the function in CFileDataFormatter
string: File Path
string:File Name
1: formatData
CAbsFormattedData Ptr
1000
CAbsDataFormatter
Either pointer to CBinaryFileFormattedData or CTextFileFormattedData
Either CBinaryFileDataFormatter or CTextFileDataFormatter

FIG. 17

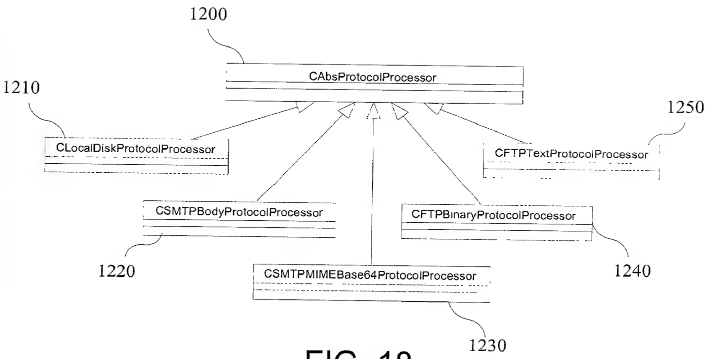

1200
CAbsProtocolProcessor
1210
CLocalDiskProtocolProcessor
1250
CFTPTextProtocolProcessor
CSMTPBodyProtocolProcessor
CFTPBinaryProtocolProcessor
1220
1240
CSMTPMIMEBase64ProtocolProcessor
1230

FIG. 18

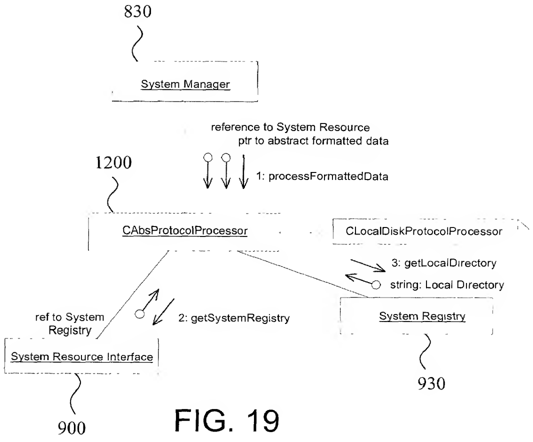

830
System Manager
reference to System Resource
ptr to abstract formatted data
1: processFormattedData
1200
CAbsProtocolProcessor
CLocalDiskProtocolProcessor
3: getLocalDirectory
string: Local Directory
ref to System Registry
2: getSystemRegistry
System Registry
System Resource Interface
930
900

FIG. 19

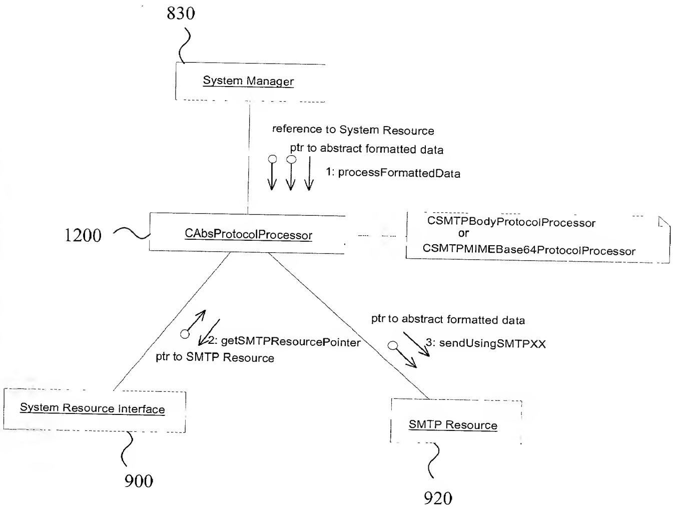
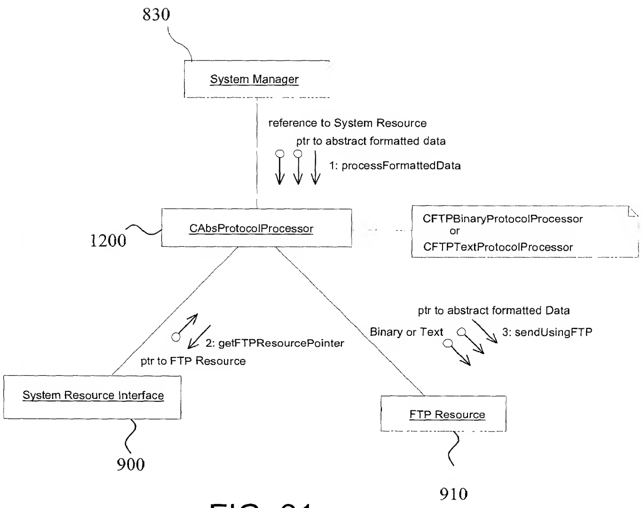

830
System Manager
reference to System Resource
ptr to abstract formatted data
1: processFormattedData
1200
CAbsProtocolProcessor
CSMTPBodyProtocolProcessor
or
CSMTPMIMEBase64ProtocolProcessor
2: getSMTPResourcePointer
ptr to SMTP Resource
ptr to abstract formatted data
3: sendUsingSMTPXX
System Resource Interface
900
SMTP Resource
920

FIG. 20

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 25 OF 32



830
System Manager
reference to System Resource
ptr to abstract formatted data
1: processFormattedData
1200
CAbsProtocolProcessor
CFTPBinaryProtocolProcessor or CFTPTextProtocolProcessor
2: getFTPResourcePointer
ptr to FTP Resource
ptr to abstract formatted Data
Binary or Text
3: sendUsingFTP
System Resource Interface
FTP Resource
900
910

FIG. 21

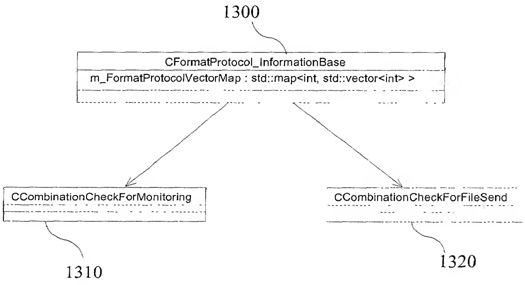

1300
CFormatProtocol_InformationBase
m_FormatProtocolVectorMap : std::map<int, std::vector<int> >
CCombinationCheckForMonitoring
CCombinationCheckForFileSend
1310
1320

FIG. 22

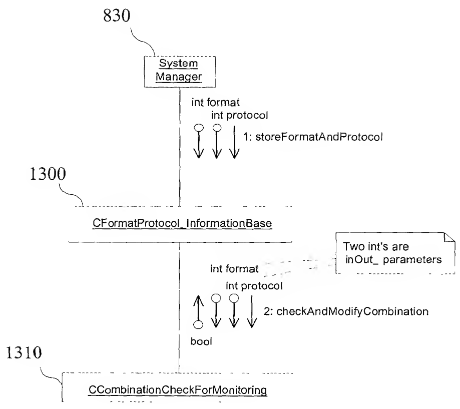

830
System Manager
int format
int protocol
1: storeFormatAndProtocol
1300
CFormatProtocol_InformationBase
Two int's are inOut_ parameters
int format
int protocol
2: checkAndModifyCombination
bool
1310
CCombinationCheckForMonitoring

FIG. 23

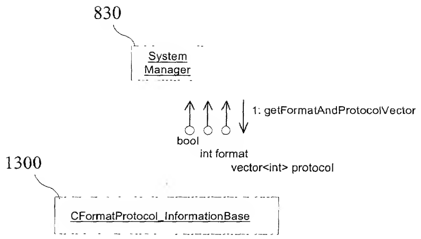

830
System Manager
1: getFormatAndProtocolVector
bool
int format
vector<int> protocol
1300
CFormatProtocol_InformationBase

FIG. 24

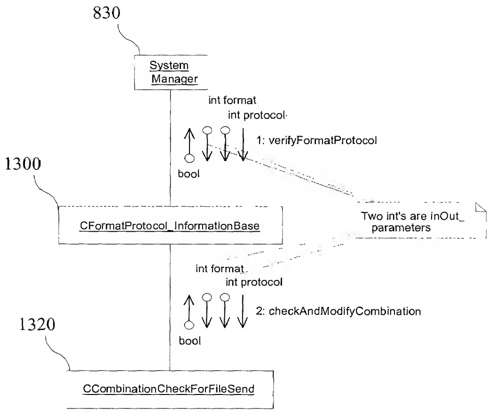

830
System Manager
int format
int protocol
1: verifyFormatProtocol
bool
1300
CFormatProtocol_InformationBase
Two int's are inOut_ parameters
int format
int protocol
2: checkAndModifyCombination
bool
1320
CCombinationCheckForFileSend

FIG. 25

Key	Value
10	{ 1 10 30 100 105 }
20	{ 1 10 30 100 105 }

FIG. 26A

1. Set return-bool to true
2. Use find function of the Map for inOut_nFormat
3. If returned iterator is end (Not found), set inOut_nFormat to the default value(10) and set return-bool to false
4. get the Set value for the key format
5. Use the find function of the Set for inOut_nProtocol
6. if returned iterator is end (Not found), set inOut_nProtocol to default (1) and set return-bool to false
7. return return-bool

FIG. 26B

OBLON, SPIVAK, ET AL
DOCKET #:194538US-2
INV: TETSURO MOTOYAMA ET AL.
SHEET 31 OF 32

09728093 .102401

Key	Value	
1	Key	Value
	1	1
	10	10
	30	30
	100	100
	105	105
5	Key	Value
	1	1
	10	30
	30	30
	100	105
	105	105

FIG. 27A

```
1.    Set return-bool = true
2.    Use find function of the Map for inOut_nFormat
3.    If returned iterator is end, set inOut_nFormat to
      the default value (5) and set return-bool to false
4.    Get the Map corresponding to the key format
5.    Use the find function of the Map for inOut_nProtocol
6.    if returned iterator is end{
          set inOut_nProtocol to default (105)
          and set return-bool to false
          }
      else {
          return-bool = (inOut_nProtocol EQ
                          (Value field corresponding to
                          inOut_nProtocol))
                         logical-AND return-bool.
          set inOut_nProtocol =
           (Value field corresponding to inOut_nProtocol).
          }
7.    return return-bool
```

FIG. 27B